

Отчет по практической работе №2

Основы работы с технологиями контейнеризации и ботами Telegram

Цель работы:

Создание сервера с постоянно работающим Telegram ботом.

Ход работы:

С помощью Windows PowerShell подключаемся к серверу-шлюзу.

```
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

PS C:\WINDOWS\system32> ssh student@193.124.118.93
The authenticity of host '193.124.118.93 (193.124.118.93)' can't be established.
ED25519 key fingerprint is SHA256:9Yff0RJ043svji7MPRcbpG4Jp3k2f7tVmDEavUEQwYQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '193.124.118.93' (ED25519) to the list of known hosts.
student@193.124.118.93's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.2.0-1015-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat Feb 15 08:46:12 2025 from 195.209.114.67
student@ruvds-x7i06:~$
```

Рис 2.1

Отсюда подключаемся к рабочему серверу.

```
student@ruvds-x7i06:~$ ssh student@10.8.0.5
student@10.8.0.5's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-131-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

12 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Sat Feb 15 17:46:16 2025 from 10.8.0.3
student@user-IPMSB-H61:~$ █
```

Рис 2.2

Создаем свой каталог по номеру зачетки и переходим в него и проверяем работу Python.

```
student@user-IPMSB-H61:~$ mkdir 220803144 && cd 220803144
student@user-IPMSB-H61:~/220803144$ python3.10
Python 3.10.16 (main, Dec  4 2024, 08:53:37) [GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Рис 2.3

Создаем и активируем окружение Python.

```
student@user-IPMSB-H61:~/220803144$ python3.10 -m venv env
```

Рис 2.4

```
student@user-IPMSB-H61:~/220803144$ source env/bin/activate
(env) student@user-IPMSB-H61:~/220803144$
```

Рис 2.5

Устанавливаем telepot.

```
(env) student@user-IPMSB-H61:~/220803144$ pip install telepot==12.7
```

Рис 2.6

Создаем нового бота через BotFather.

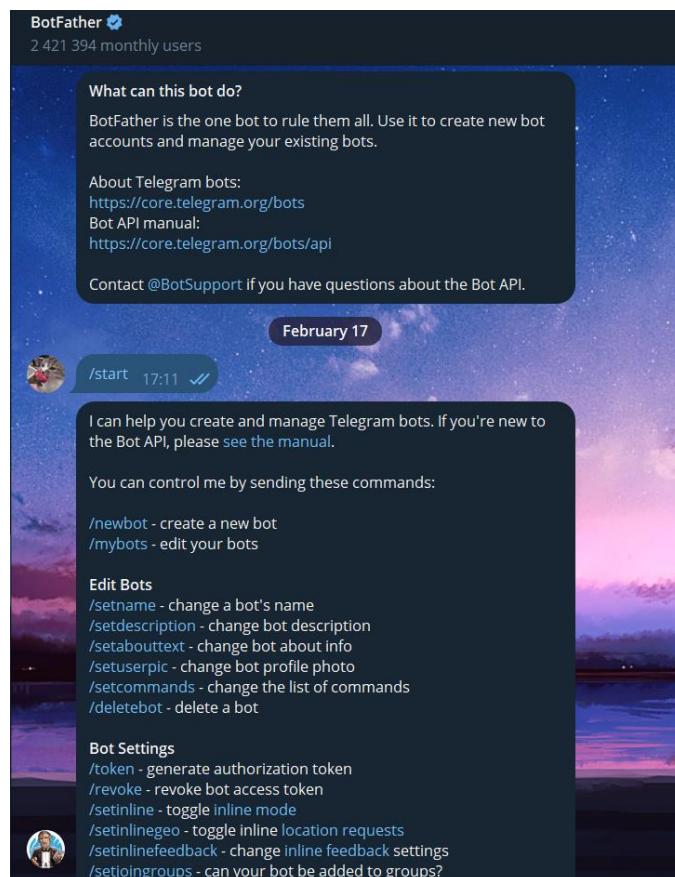


Рис 2.7

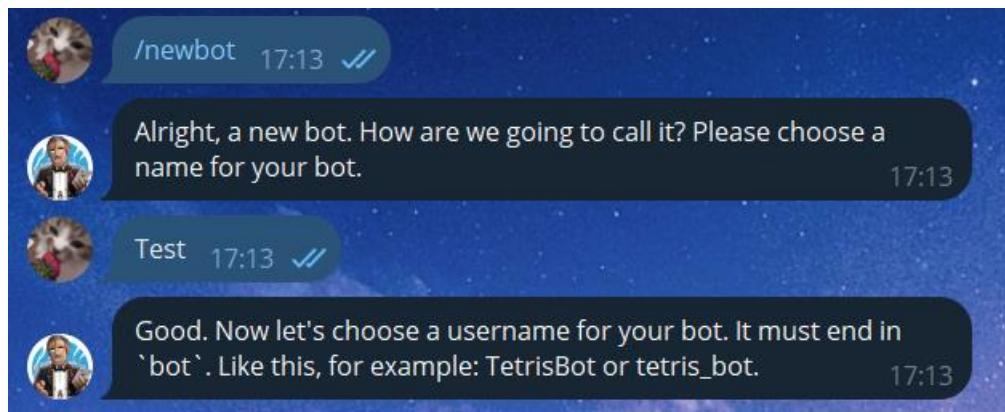


Рис 2.8

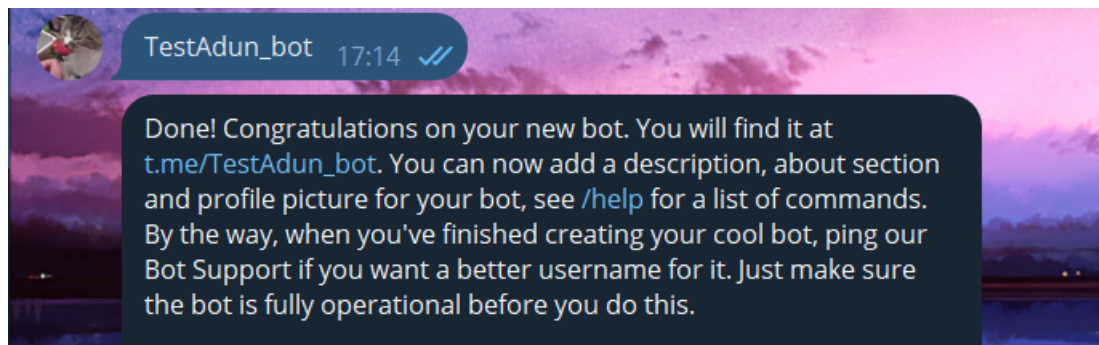


Рис 2.9

Запускаем текстовый редактор и вставляем код программы, заменив токен на свой.

```

student@user-IPMSB-H61: ~/220803144
GNU nano 4.8 bot.py
import telepot
import time
def handle(msg):
    chat_id = msg['chat']['id']
    command = msg['text']
    print('Got command: %s' % command)
    print('From : %s' % chat_id)
    if command == '/command1':
        bot.sendMessage(chat_id, 'Oks')
    elif command == '/command2':
        bot.sendMessage(chat_id, 'Ok')
bot = telepot.Bot('***** PUT YOUR TOKEN HERE *****')
bot.message_loop(handle)
print('I am listening ...')
while 1:
    time.sleep(10)

```

Рис 2.10

Запускаем программу и проверяем работу бота.

```
(env) student@user-IPMSB-H61:~/220803144$ python bot.py
I am listening ...
Got command: /start
From : 1017881787
Got command: /command1
From : 1017881787
Got command: /command2
From : 1017881787
```

Рис 2.11

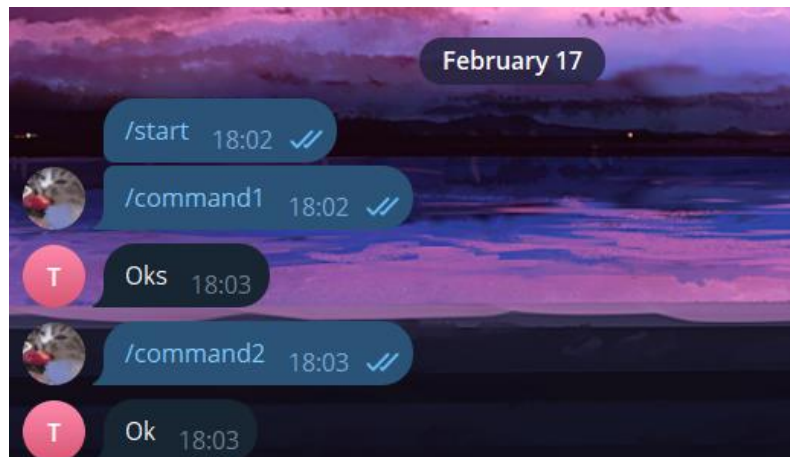


Рис 2.12

Отключаем python env и создаем файл requirements со списком библиотек, необходимых для работы программы.

```
(env) student@user-IPMSB-H61:~/220803144$ deactivate
student@user-IPMSB-H61:~/220803144$ nano requirements.txt
```

Рис 2.13

Создаем файл для сборки docker образа и собираем его.

```
student@user-IPMSB-H61:~/220803144$ nano Dockerfile
```

Рис 2.14

```
student@user-IPMSB-H61:~/220803144$ docker buildx build -t 220803144 .
[+] Building 7.8s (13/13) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 295B
=> [internal] load metadata for docker.io/library/python:3.10
=> [internal] load metadata for docker.io/library/python:3.10-slim
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [builder 1/3] FROM docker.io/library/python:3.10@sha256:76722e4ce53774c1f5eb0ba145ed657b908e7aa129fee75eca69b
=> [stage-1 1/4] FROM docker.io/library/python:3.10-slim@sha256:6baa080231f011c500e196ae01526a7175f058672498196
=> [internal] load build context
=> => transferring context: 527B
=> CACHED [stage-1 2/4] WORKDIR /code
=> CACHED [builder 2/3] COPY requirements.txt
=> CACHED [builder 3/3] RUN pip install --user -r requirements.txt
=> CACHED [stage-1 3/4] COPY --from=builder /root/.local /root/.local
=> [stage-1 4/4] COPY ./bot.py
=> exporting to image
=> => exporting layers
=> => writing image sha256:7f46e5bf23a7ed701379bf98096ed3e18af5c5c113576a82b6d7dcafa6d1f6
=> => naming to docker.io/library/220803144
```

Рис 2.15

Запускаем docker образ в режиме работы в фоне. Получаем CONTAINER ID.

```
student@user-IPMSB-H61:~/220803144$ docker run -d --restart=always 220803144  
0d9c445b5b397667aaa8936bb76bab2994958157d4065e56ee974163f16b6240
```

Рис 2.16

Используем CONTAINER ID для просмотра log.

```
student@user-IPMSB-H61:~/220803144$ docker logs 0d9c445b5b397667aaa8936bb76bab2994958157d4065e56ee974163f16b6240  
I am listening ...  
Got command: /command1  
From : 1017881787
```

Рис 2.17

Сохраняем docker image в виде архива.

```
student@user-IPMSB-H61:~/220803144$ docker save -o ./docker_image_220803144.tar 220803144
```

Рис 2.18

Отключаемся от рабочего сервера и копируем архив docker-образа на шлюз.

```
student@ruvds-x7i06:~$ scp student@10.8.0.5:/home/student/220803144/docker_image_220803144.tar .  
student@10.8.0.5's password:  
docker_image_220803144.tar 100% 138MB 1.9MB/s 01:11  
student@ruvds-x7i06:~$
```

Рис 2.19

Отключаемся от шлюз-сервера и копируем файл архива к себе на ПК.

```
PS C:\WINDOWS\system32> scp student@193.124.118.93:/home/student/docker_image_220803144.tar .  
student@193.124.118.93's password:  
docker_image_220803144.tar 100% 138MB 1.8MB/s 01:17  
PS C:\WINDOWS\system32>
```

Рис 2.20

Вывод: В результате практической были изучены основы работы с технологиями контейнеризации и ботами Telegram. Был создан Telegram-бот, который отвечает на две команды, демонстрируя базовые возможности взаимодействия с пользователем. Для реализации проекта использовалась технология Docker, что позволило создать и управлять изолированными средами, обеспечивая легкость развертывания и масштабирования приложения.